Neural Sequence Generation with Constraints via Beam Search with Cuts: A Case Study on VRP

POUYA SHATI, ELDAN COHEN, SHEILA MCILRAITH

SOCS-24

UNIVERSITY OF TORONTO

VECTOR INSTITUTE FOR ARTIFICIAL INTELLIGENCE

JUNE ‘24

Motivation

- **Neural sequence generation** can successfully solve **combinatorial optimization problems**
  - However, it does not support **hard requirements**
  - **Beam search**, as an agnostic approach, lacks guarantee even at large quantities

- **Vehicle routing problems (VRP)** are used as the case study
  - They are solved using neural sequence models employing **transformers** and **RL**
  - **Global constraints** that require meticulous reasoning are absent

Our Contribution

- **Beam search with cuts (BSC)**,
  - A **modular** framework combining any **pre-trained** neural sequence model with **requirements**
  - **Requirements** represent a **set of constraints** that solutions must satisfy

- Requirements encoded as **constraint satisfaction problems (CSP)**
  - **Bin packing** encoded in **IP**
  - **Regular language** specification encoded in **SAT**
  - Solve **3 VRP variants**

- Experimental results showing that BSC
  - Satisfies requirements with **negligible cost** to quality
  - **Scales exponentially better** when problem size increases

Background

Vehicle Routing Problems

Beam search with cuts

Experiments

- Sequence Generation with Requirements
- Constraint Satisfaction Problems

Sequence Generation

- **Tokens**: Σ
- A sequence of tokens as the **solution**: $x \in \Sigma^*$

Sequence Generation

- **Tokens**: Σ
- A sequence of tokens as the **solution**: $x \in \Sigma^*$

- **Next token prediction** function using **neural model**: $p: \Sigma^* \rightarrow \mathcal{P}(\Sigma)$
- Sequence **score**: $\theta(x) = \Pi_i \, p(x_1, x_2, \ldots, x_i)[x_{i+1}]$

Sequence Generation

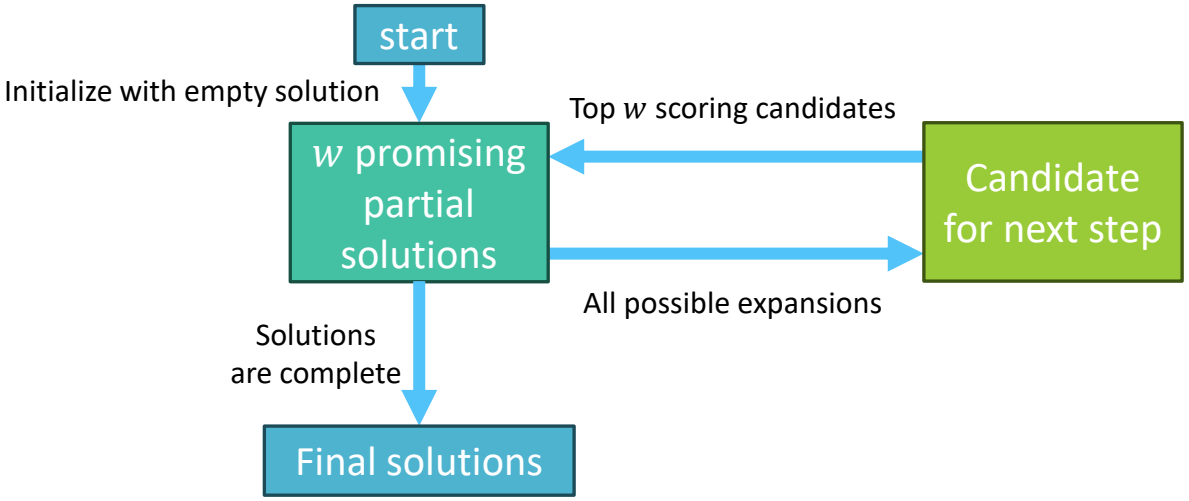
- **Tokens**: Σ
- A sequence of tokens as the **solution**: $x \in \Sigma^*$

- **Next token prediction** function using **neural model**: $p: \Sigma^* \rightarrow \mathcal{P}(\Sigma)$
- Sequence **score**: $\theta(x) = \Pi_i \, p(x_1, x_2, \ldots, x_i)[x_{i+1}]$

- **Beam search** decoder
  - Sets of **partial solutions** of size i: S_i
  - $S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma\})$

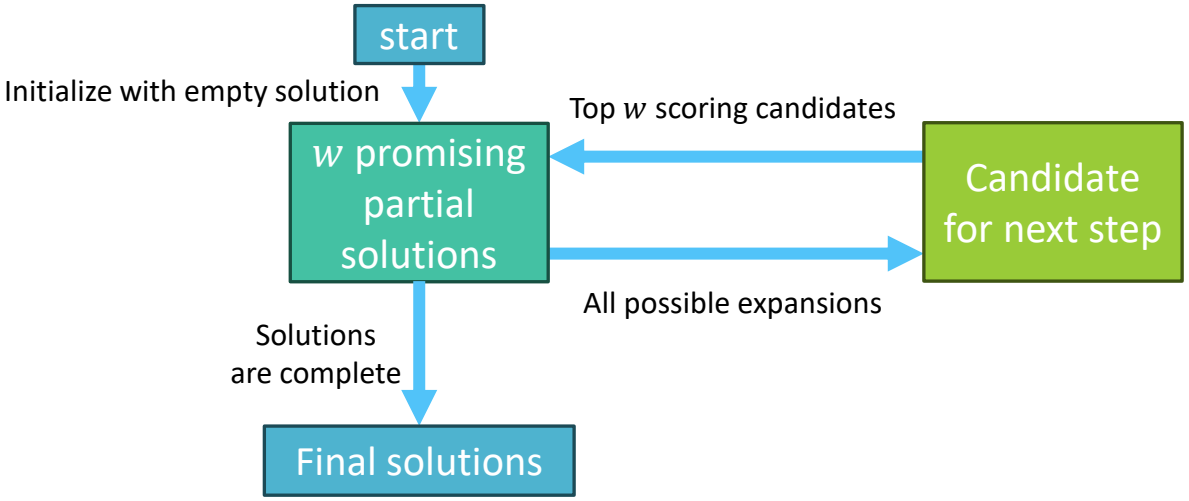
Beam Search

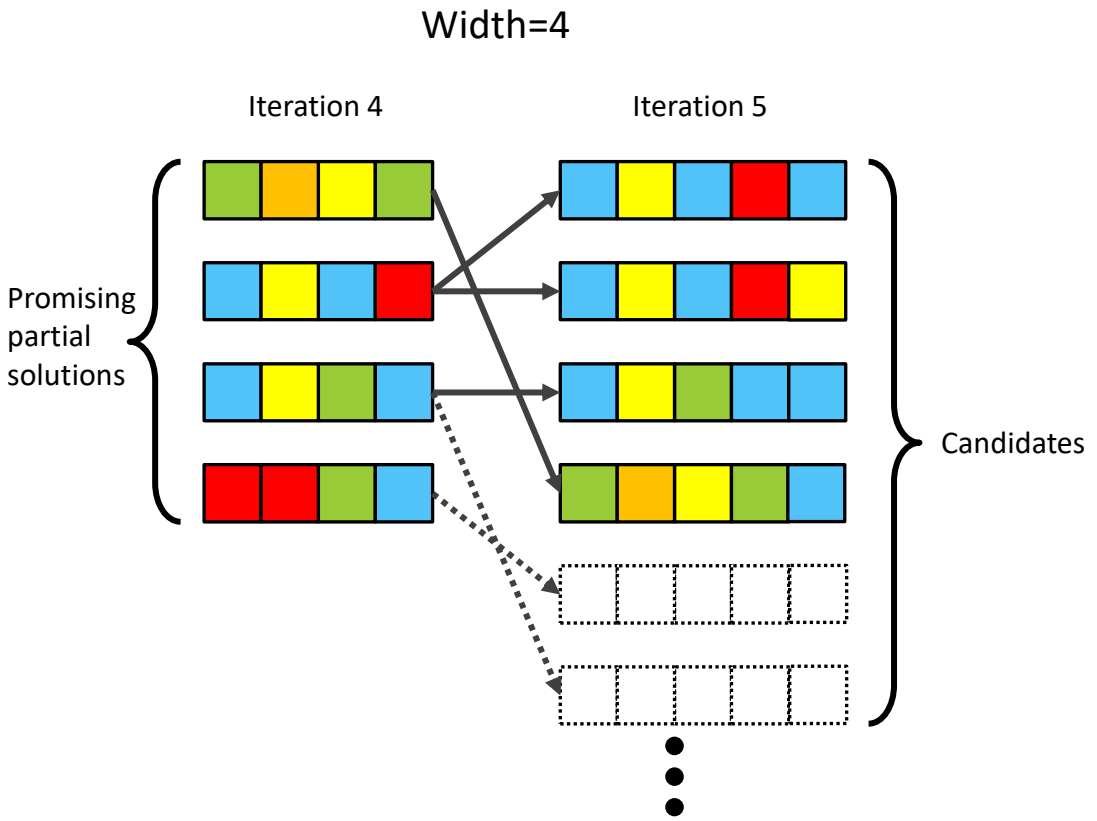
- **Beam search** decoder
  - Sets of **partial solutions** of size i: S_i
  - Beam **width** ($\boldsymbol{w}$): number of partial solutions
  - $S_i = argmax_{1:w}(\{\theta(x.a)|x \in S_{i-1}, a \in \Sigma\})$



Beam Search

- **Beam search** decoder
  - Sets of **partial solutions** of size i: S_i
  - Beam **width** (w): number of partial solutions
  - $S_i = argmax_{1:w}(\{\theta(x.a)|x \in S_{i-1}, a \in \Sigma\})$





Sequence Generation with Requirements

- **Tokens**: Σ
- A sequence of tokens as the **solution**: $x \in \Sigma^*$

- **Requirement**: $R \subseteq \Sigma^*$
  - Refers to sequences satisfying a set of constraints

- **Next token prediction** function using **neural model**: $p: \Sigma^* \rightarrow \mathcal{P}(\Sigma)$
- Sequence **score**: $\theta(x) = \Pi_i \, p(x_1, x_2, \ldots, x_i)[x_{i+1}]$

- **Beam search** decoder
  - Sets of **partial solutions** of size i: S_i
  - $S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma\})$

- Final solutions: $S_k \cap R$
  - Is agnostic of the requirement and lacks guarantee

Constraint Satisfaction Problems

- CSP:
  - A finite set of **variables** with corresponding **domains**
  - A finite set of **constraints** on variables
  - Solution, **a satisfying assignment** to all variables

Constraint Satisfaction Problems

- CSP:
  - A finite set of **variables** with corresponding **domains**
  - A finite set of **constraints** on variables
  - Solution, **a satisfying assignment** to all variables

- **SAT**:
  - Boolean domains
  - Disjunctive clauses on literals

$$(x_1 \vee \neg x_2 \vee x_4)$$

- **Integer programming (IP)**:
  - Integer domains
  - Linear constraints

$$x_1 + 2x_2 - 4.5x_3 \geq 5$$

Background

Vehicle Routing Problems

Beam search with cuts

Experiments

- VRP Variants
- VRP Neural Sequence Model

VRP Variants

- VRP variants mostly involve:
  - Navigating **vehicles** through **nodes**
  - Commonly include **capacity** constraints and optimize **distance**

VRP Variants

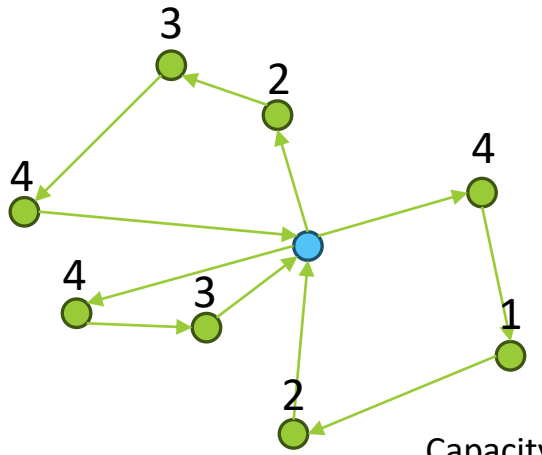
- **Nodes:** $N = \{n_i | n_i \in \mathbb{R} \times \mathbb{R}\}$
- **Objective:** minimize total distance

VRP Variants

- **Nodes:** $N = \{n_i | n_i \in \mathbb{R} \times \mathbb{R}\}$
- **Objective:** minimize total distance

- Constrained Vehicle Routing Problem with Maximum Tours **(CVRPM)**:
  - **Demand** function: $D: N \rightarrow \mathbb{N}$
  - **Depot** node: n_d
  - **Capacity**: $c \in \mathbb{N}$
  - Maximum number of **tours**: $m \in \mathbb{N}$

  - Solution: a series of **tours** $\boldsymbol{T}$ partitioning the nodes that respects the capacity



Capacity: 10
Tours: 3

VRP Neural Sequence Model

- Baseline: **Kool et al. [1]**
  - Appeared at **ICLR ’19**

- Uses a **deep learning** model
  - Based on **attention** layers
  - Trained using **REINFORCE [2]**
  - Uses **nodes** as tokens and minimizes **distance**

- Can be used to solve **CVRPM**
  - Supports **CVRP** directly
  - **Beam search** with large width value to satisfy **max tours** requirements

Background

Vehicle Routing Problems

Beam search with cuts

Experiments

- BSC Decoder
- Bin Packing Requirement
- Regular Language Requirements

BSC Decoder

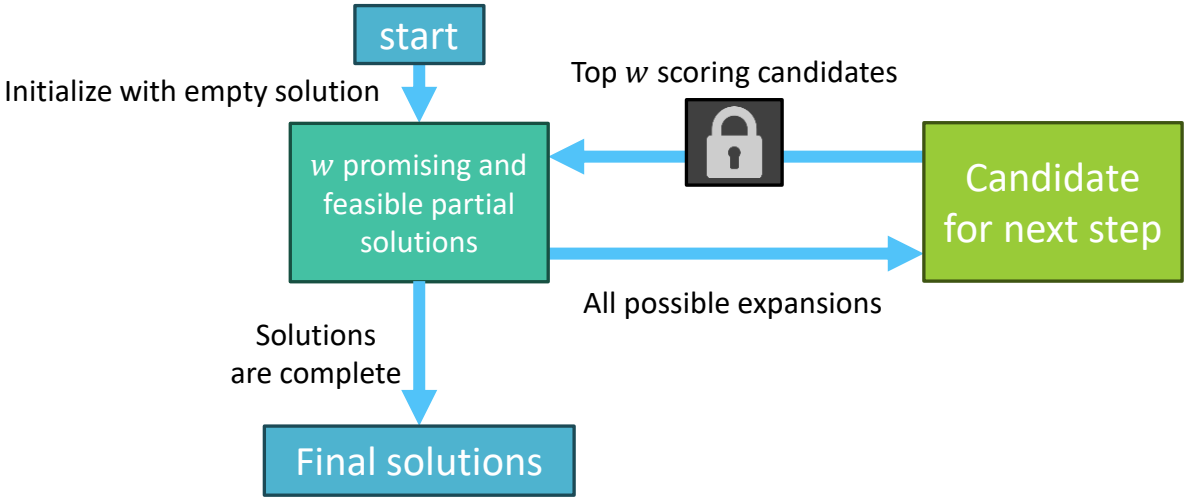
- Beam search that employs **cuts**:
  - Explicitly checks whether a **partial solution** can be **extended** to a **complete feasible one**
  - **Impedes infeasible** partial solutions from expanding further

$$S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma, \exists x' : x.a.x' \in R \wedge [x.a.x' \; is \; complete] \})$$

BSC Decoder

- Beam search that employs **cuts**:

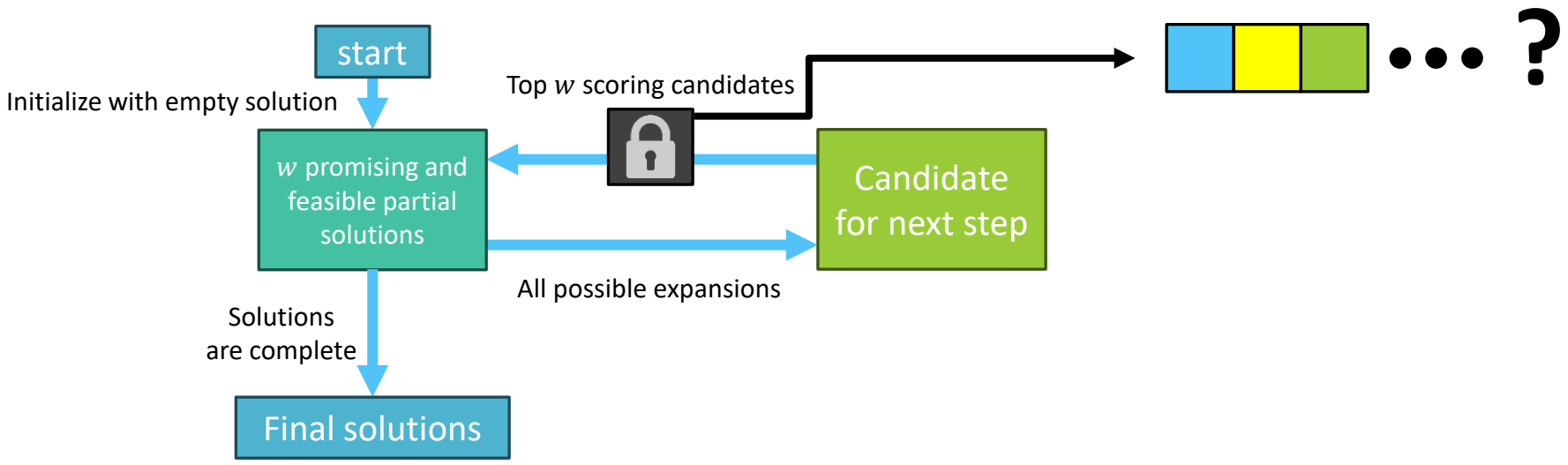
$$S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma, \exists x' : x.a.x' \in R \wedge [x.a.x' \; is \; complete]\})$$



BSC Decoder

- Beam search that employs **cuts**:

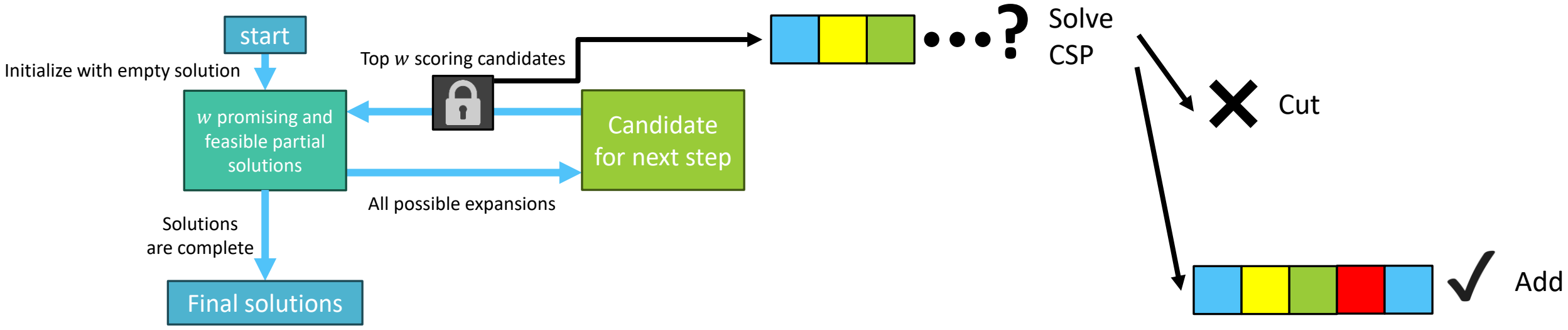
$$S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma, \exists x' : x.a.x' \in R \wedge [x.a.x' \; is \; complete]\})$$



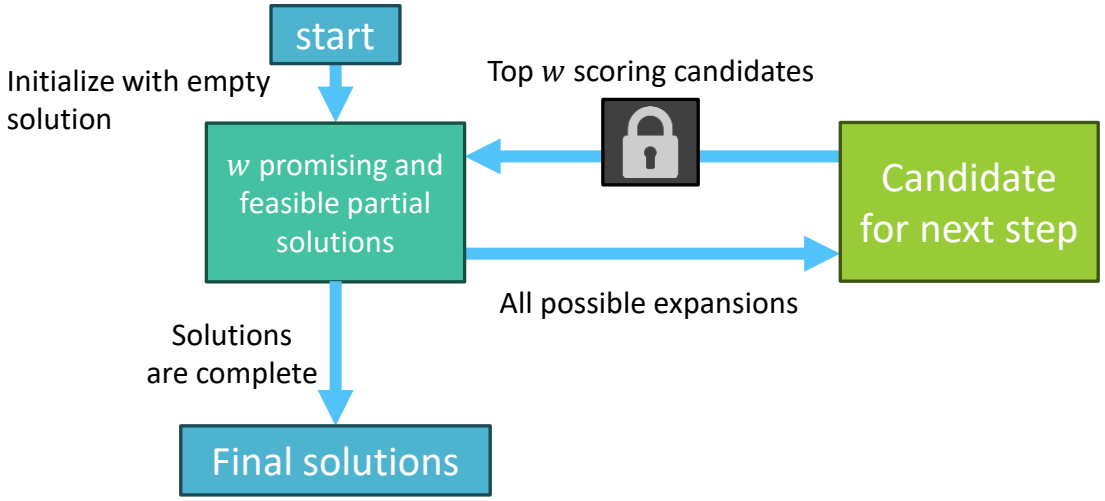
BSC Decoder

- Beam search that employs **cuts**:

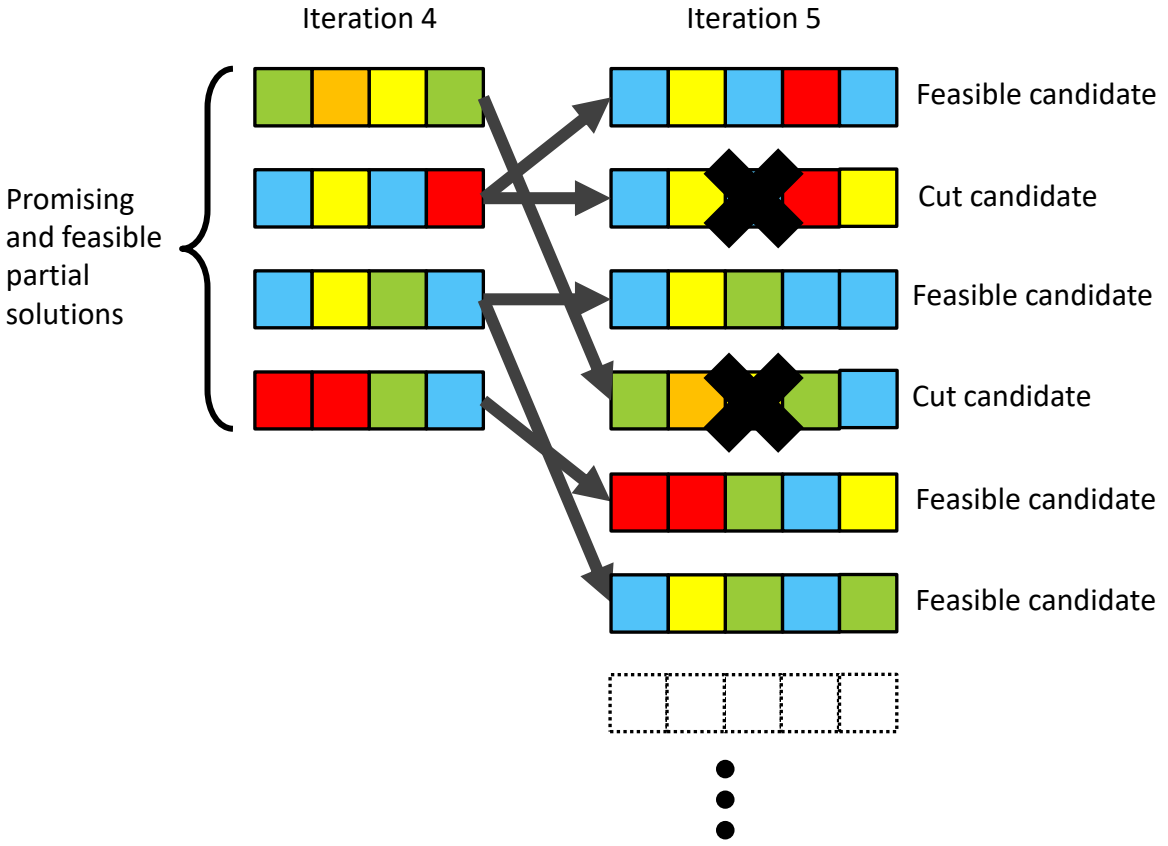
$$S_i = argmax_{1:w}(\{\theta(x.a) | x \in S_{i-1}, a \in \Sigma, \exists x' : x.a.x' \in R \wedge [x.a.x' \; is \; complete]\})$$



BSC Decoder



Width=4



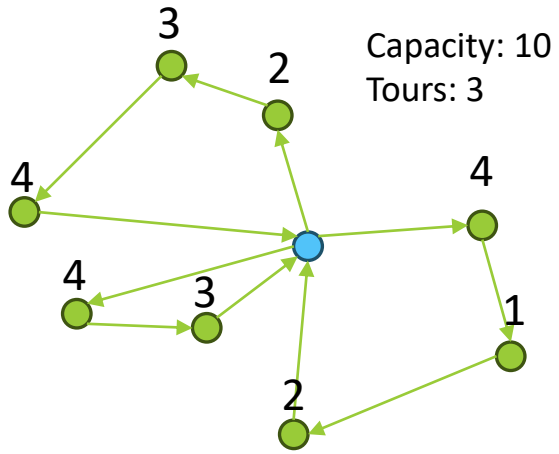
Bin Packing Requirement

- Definition:
  - Set of **items**: I
  - **Weights**: $W: I \rightarrow \mathbb{N}$
  - Bin **capacity**: $c \in \mathbb{N}$
  - **Number** of bins: $m \in \mathbb{N}$

  - Solution: a **partition of items** $\boldsymbol{B = \{B_1, B_2, \ldots, B_m\}}$ that respects the capacity

Bin Packing Requirement

- Application:
  - Combined with CVRP to solve **CVRPM**:
    - Bins = Tours
    - Items = Nodes
    - Weights = Demands



Bin Packing Requirement

- Encoding in **IP**:
  - Variable $\boldsymbol{a_{i,j}}$ represents **item $\boldsymbol{i}$** being assigned to **bin $\boldsymbol{j}$**

- Adherence to the partial solution:
  - **Fixed assignments** for $B_1^F, B_2^F, \dots, B_{t^F+1}^F$
  - With $B_1^F, B_2^F, \dots, B_{t^F}^F$ **closed off**

$$\forall i \in I : \sum\nolimits_j a_{i,j} = 1 \tag{1}$$

$$\forall j \in [1, m] : \sum\nolimits_i a_{i,j} W(i) \leq c \tag{2}$$

$$\forall j \leq t^F + 1, i \in B_j^F : a_{i,j} = 1 \tag{3}$$

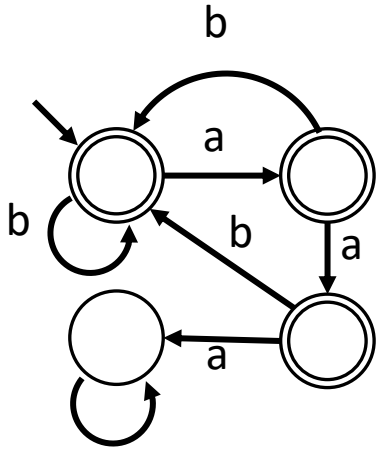
$$\forall j \leq t^F, i \notin \bigcup\nolimits_t B_t^F : a_{i,j} = 0 \tag{4}$$

Regular Language Requirement

- Definition:
  - **DFA**: $\mathcal{A}$

    Alphabet: $\Sigma_{\mathcal{A}}$, Set of states: $Q_{\mathcal{A}}$, Initial state: $q_0 \in Q_{\mathcal{A}}$,

    Final states: $Q_{\mathcal{A}}^{F} \subseteq Q_{\mathcal{A}}$, Transition function: $\delta_{\mathcal{A}}: Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}} \to Q_{\mathcal{A}}$
  - Possible **inputs**: $W_{\mathcal{A}} \subseteq \Sigma_{\mathcal{A}}^{*}$
  - Solution: $\boldsymbol{w \in W_{\mathcal{A}}}$ **with** $\boldsymbol{\Delta(q_0, w) \in Q_{\mathcal{A}}^{F}}$ where
    - $\Delta(q, w) = \begin{cases} \Delta(\delta(q, a), w'), & a \in \Sigma_{\mathcal{A}}, w = a.w' \\ q, & w = \epsilon \end{cases}$



Regular Language Requirement

- Encoding in SAT:
  - Variable $\boldsymbol{d_{i,a}}$ represents that $\boldsymbol{w_i = a}$
  - Variable $\boldsymbol{s_{i,q}}$ represents that DFA is in **state** $\boldsymbol{q}$ at **step** $\boldsymbol{i}$
  - Constant $\boldsymbol{W_a}$ represent **count of** $\boldsymbol{a}$ in $\boldsymbol{\sigma(N)}$

- Adherence to the partial solution:
  - **Fixed assignments** based on w_i^F for $1..l^F$

- **Disjunctive** clauses (… ∨ …)
- **Cardinality** clauses … ≤ …
- **Assumptions** […]

$$\forall i : (\bigvee\nolimits_a d_{i,a}) \tag{5}$$
$$\forall i : \sum\nolimits_a d_{i,a} \leq 1 \tag{6}$$
$$\forall a : \sum\nolimits_i d_{i,a} \leq W_a \tag{7}$$
$$\forall a : \sum\nolimits_i \neg d_{i,a} \leq |N| - W_a \tag{8}$$
$$\forall i : (\bigvee\nolimits_q s_{i,q}) \tag{9}$$
$$\forall i : \sum\nolimits_q s_{i,q} \leq 1 \tag{10}$$
$$\forall a : (\neg d_{1,a} \vee s_{1,\delta(q_0,a)}) \tag{11}$$
$$\forall i > 1, q, a : (\neg s_{i-1,q} \vee \neg d_{i,a} \vee s_{i,\delta(q,a)}) \tag{12}$$
$$(\bigvee\nolimits_{q \in Q_{\mathcal{A}}^F} s_{|N|,q}) \tag{13}$$
$$\forall i \leq l^F : [d_{i,w_i^F}] \tag{14}$$

Background

Vehicle Routing Problems

Beam search with cuts

Experiments

- Experimental Setup
- Sequence Generation with Requirements
- Tightening Requirements
- Scaling Problem Size

Experimental Setup

- Solvers:
  - IP: **Gurobi**
  - SAT: **Gluecard 4**
- Timeout limit: **10** seconds for each CSP call
- Datasets:
  - **Uchoa et al. [3]**
  - **Synthetic,** following Kool et al. [1]

Sequence Generation with Requirements

- Solved **CVRPM** for **Uchoa et al. [3]** instances (with $m \leq 20$)

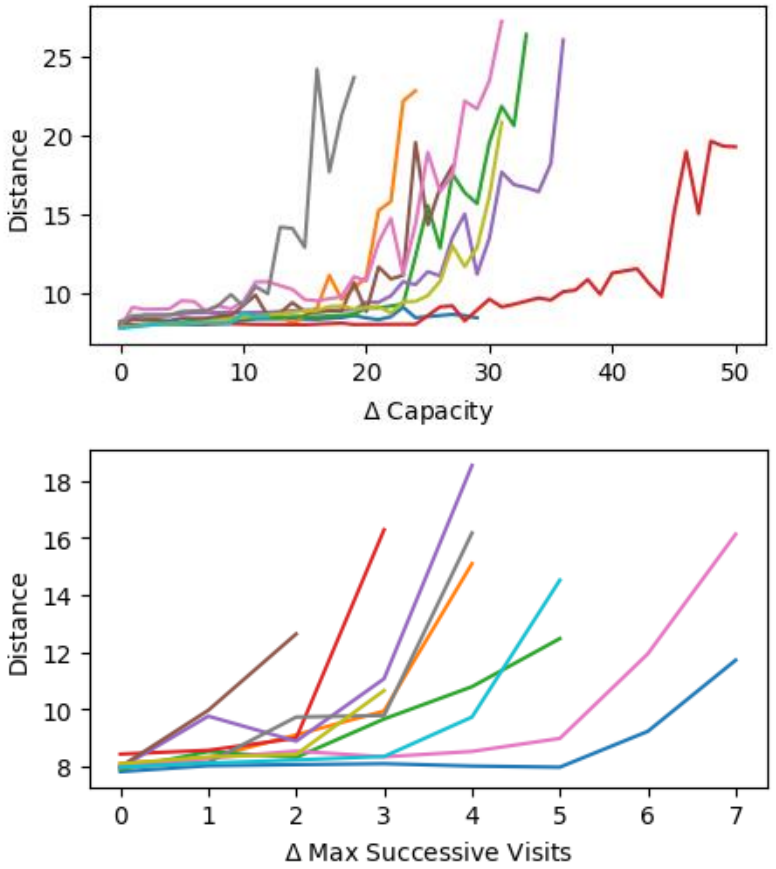
- Compared **BS** (width=8096) against **BSC** (width=4)
  - On **9 out of 27** instances where BS failed

- Showed in results that **BSC**
  - Satisfies **requirement**
  - Causes negligible cost to **quality**
  - Operates on **smaller width** and takes **less runtime**

Instance ($\|N\|$,m)	Beam Search Time (s)	Beam Search m	Beam Search with Cuts Time (s)	Beam Search with Cuts m	Beam Search with Cuts Δdis.	Beam Search with Cuts Cuts
(134,13)	26.7	14	45.1	13	0.8%	168
(157,13)	35.7	14	14.9	13	-12.2%	17
(190,8)	49.3	11	14.4	8	-23.9%	52
(209,16)	60.7	17	28.3	16	2.0%	19
(214,11)	61.7	12	27.0	11	5.7%	92
(233,16)	73.4	17	384.6	16	20.1%	219
(256,16)	89.3	17	630.4	16	9.0%	408
(367,17)	185.6	18	84.4	17	-0.6%	6
(411,19)	236.7	22	116.7	19	-14.6%	44

Tightening Requirements

- Solved **TSPD** and **TSPR** with **incrementally tighter** requirements
  - Used **synthetic** datasets for both problems
- Compared **BS** ($\Delta = 0$, width=8096) vs **BSC** ($\Delta > 0$, width=4)
- Showed in results that **BSC**
  - Can **tighten requirements** with negligible cost to quality
  - Produces **unstable results** for requirements that are too tight
  - Shows quality and tightness trade-off until **infeasibility**



Scaling Problem Size

- **BS** is intuitively more likely to violate the requirement as size increases
- Solved **TSPR** with a **quantifiable** requirement for synthetic instances
- Compared **BSC** (width=4) vs complete **BS**
  - Recorded the **first width** value resulting in satisfaction
- Showed in results that **BSC** scales **exponentially** better
  - As **requirements** are **strengthened**
  - As **solution length** is **increased**

$\|N\|$	R	Beam Search $log(w)$	Beam Search with Cuts Cuts	Beam Search with Cuts Time (s)
24	P_1	2.7	2.8	1.81
	P_2	4.4	6.4	1.79
	P_3	6.8	8.4	1.80
	P_4	8.9	12.4	1.78
	P_5	10.5	15.4	1.87
12	P_3	1.2	3.5	1.73
24		6.4	8.4	1.80
36		11.1	16.7	1.95

Summary

- Beam search with cuts
  - **VRP** neural sequence model
  - **Bin packing** requirement encoded in **IP**
  - **Regular language** requirement encoded in **SAT**
  - Solved **CVRPM** variant

- Requirements **applicable** in other settings

- Experiments
  - Satisfaction of hard **requirements** with **negligible cost** to quality
  - Trade-off between requirement **tightness** and **quality** until **infeasibility**
  - **Exponentially** better scaling in larger problems

Our Paper Has More!

- Solved **TSPD** using **Bin Packing** requirement
- Solved **TSPR** using **Regular language** requirement
- Experiments:
  - **Incremental** solving of CSPs
  - **Sub-width,** a hybrid approach

Future Work

- **Equivalency checks** between partial solutions
  - To cache and query **feasibility** results
  - To cut strictly **worse solutions**
  - To increase solution **diversity**

- Applications to **other neural sequence models**
  - Planning
  - Program synthesis

- New types of **requirements**

- Integration with **Beam-Stack** to enable **backtracks**

Neural Sequence Generation with Constraints via Beam Search with Cuts: A Case Study on VRP

Thank you for your time!
Q & A

POUYA SHATI, ELDAN COHEN, SHEILA MCILRAITH

UNIVERSITY OF TORONTO

VECTOR INSTITUTE FOR ARTIFICIAL INTELLIGENCE

SOCS-24

JUNE '24

[1] Kool, W.; van Hoof, H.; and Welling, M. 2018. Attention, Learn to Solve Routing Problems! In ICLR.

[2] Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning.

[3] Uchoa, E.; Pecin, D.; Pessoa, A.; Poggi, M.; Vidal, T.; and Subramanian, A. 2017. New benchmark instances for the capacitated vehicle routing problem. EJOR.

Poster:
Saturday 8th, Session 2